EXHIBIT A to

USA Motion to Exclude Any Testimony From Erik Min That He Would Base on Late-Disclosed Documents

CrossVerify

Protects, Verifies, and Permissions

Your Personal Digital Identity

FOR INTERNAL USE ONLY

Functional and Technical Overview

07 November 2017

Contents

Document Control

Version	Date	Contributor	Change
0.1	07 November 2017	Nigel Quantick	Consolidation of all prior CV docs

Introduction

CrossVerify is a platform that enables companies to digitally manage and authenticate user identities. It creates ecosystems around each legal entity whilst putting end users are the heart of those systems. It does this in such a way that it protects and verifies end users, and they can permission usage of their personal digital identity to one or more legal entities.

CrossVerify achieves this by combining the world's leading and state of the art biometric software from Aware Inc. with CrossVerify's patent pending technology that encrypts and stores the data in an immutable and secure record on a private blockchain. Aware's technology is used by most of the world's leading national security and border force agencies.

The business model is currently configured so that a user is invited to enrol with CrossVerify. It is anticipated that the model will pivot to a self enrolment model in a future version when users are expected to take control of their own digital identity. Until that point the global business model is still the 'old world' version where governments and corporations maintain users' identities.

Use of this overview

A functional overview is given of the solution from registering a new client through to the roadmap, along with the technology stack and technical specifications of the servers in the Appendix.

Company Registration

Once a company has commercially signed up for the CrossVerify service the CrossVerify Administrator will add them to the system which initiates email to the company Super Administrator.

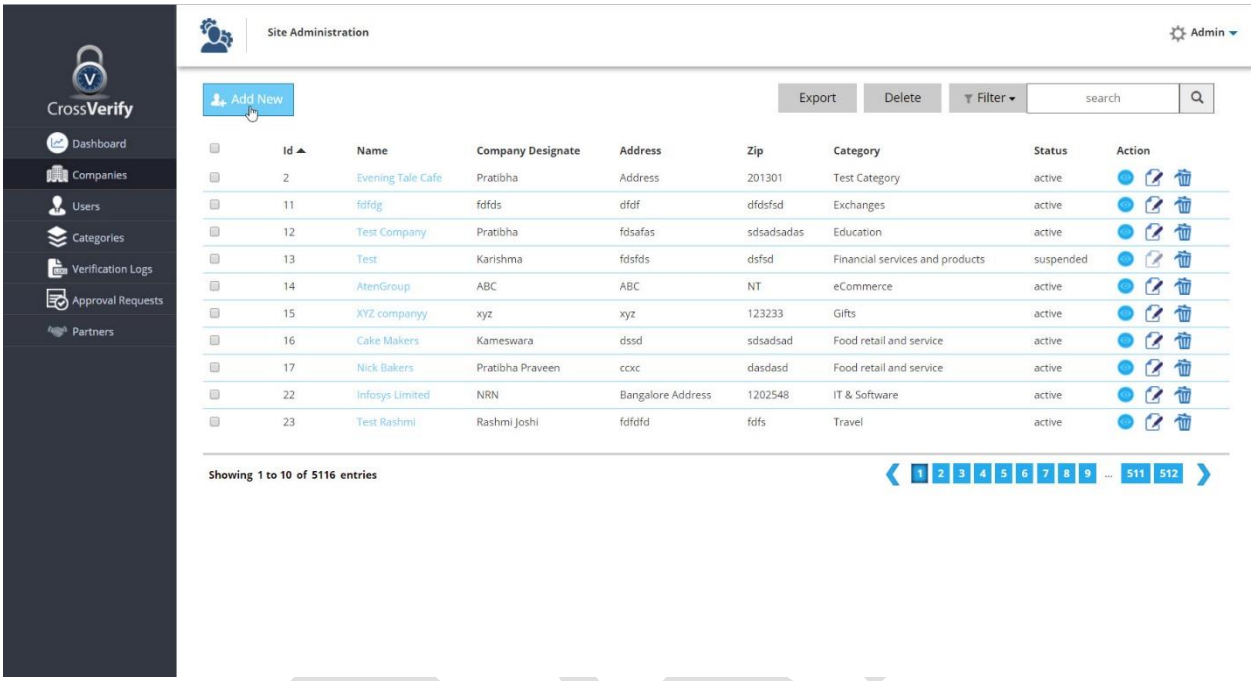


Figure 1: CrossVerify administrator Companies view

Initial Client Registration

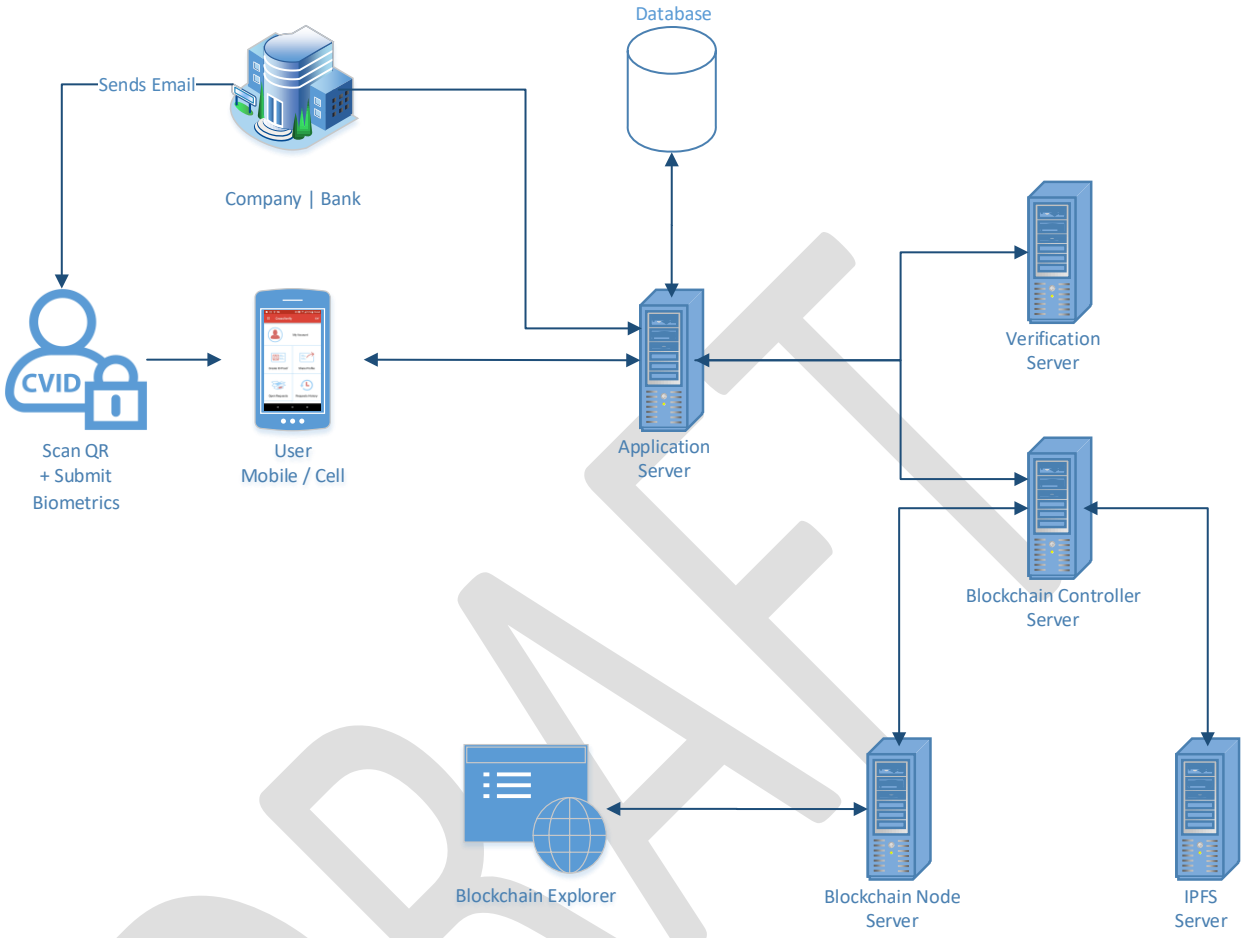
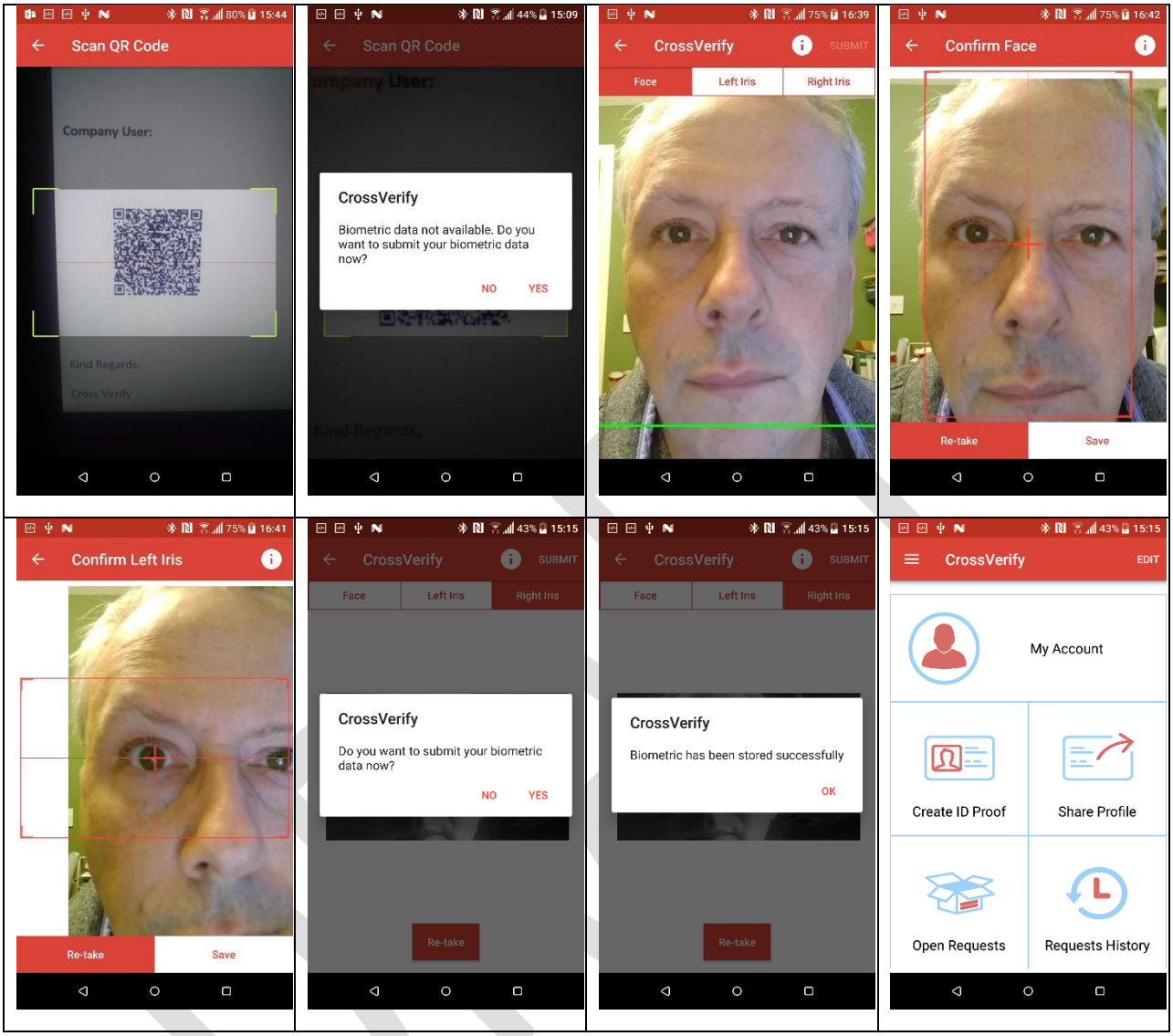


Figure 2 Initial Client Registration

Company

1) User registers with a Company.
2) The company is a trusted source and uploads client some personal data, e.g. name, email ID etc.
3) Company can upload individually or in batch mode the following file types: .jpg, .jpeg, .gif, and .png.
4) The user receives a welcome email on their registered email-ID.
5) The user is prompted to download the CrossVerify mobile app from either the Apple or Google App stores.
6) After opening the CrossVerify app, the first step for the user is to scan the QR code from the welcome email and they are then prompted to submit their biometric data.

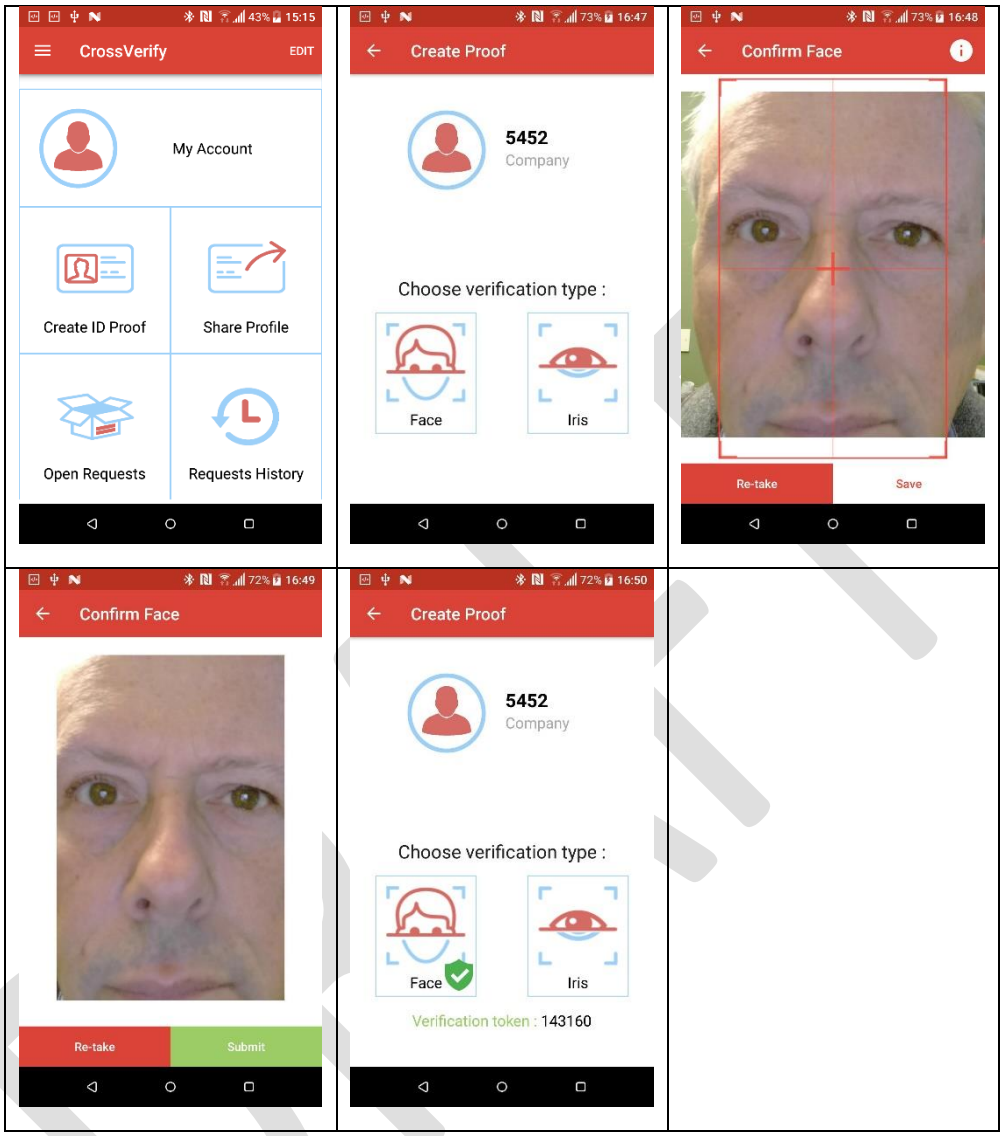
User sign-up journey



Mobile App (Push) After registration has already been done

7) The user is asked by the bank or company to confirm their identification. The user opens the mobile app and presses the Create Proof ID button.
8) The user is told to do take their biometric data, then the data is pushed over to the application server.

User verification journey



Technical overview

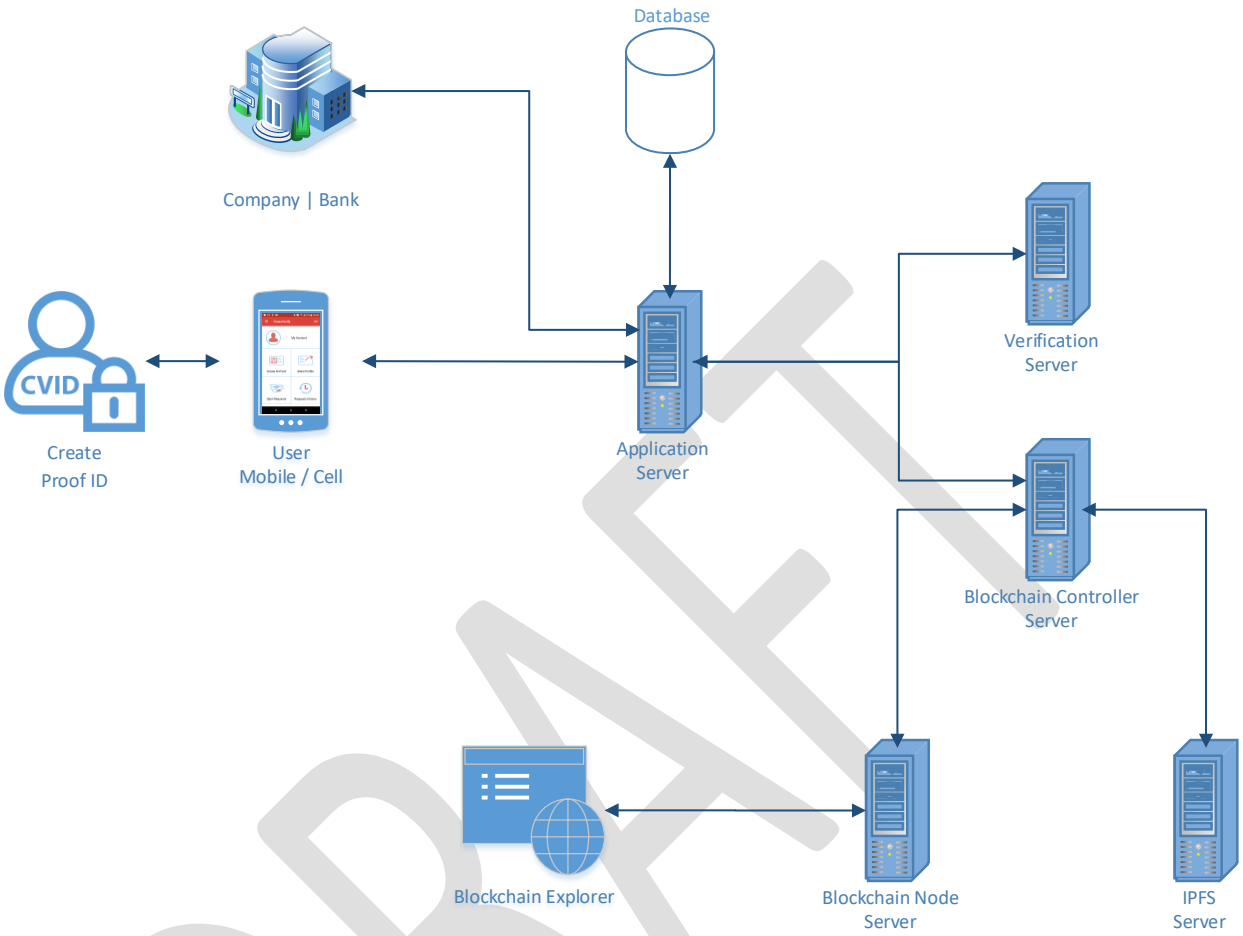


Figure 3 Technical Overview of the Proof ID

Application Server

9) The application server interacts with the verification server & the blockchain controller.
10) When the application server receives a request to confirm a person's biometrics, the application server receives the information from the mobile application. It then pulls the user's registration biometric data from the blockchain controller.
11) The application server then sends the biometric data from the mobile application and the blockchain controller over to the verification server for matching.

Blockchain Explorer

12) The blockchain explorer displays all the transaction hashes and addresses which have taken place on the blockchain node server.
13) The transaction takes place whenever data is pushed to IPFS for example in the case of the creation or update of user details. This transaction history will be maintained on the blockchain.

14) The blockchain explorer reads all of the information on the blockchain server.

    Please note: as per the original technical design, the concept of public/private keys has not been implemented for the explorer as this was not required.

    CV-IDs are the unique identity of each user that are assigned at the time of creation of that user and this CV-ID is the wallet address of the blockchain.

    Details of transaction history contain such fields as: cv_id_of_user, primary_upload_id_of_user etc.

Blockchain Node Server

15) Creates The CV-ID, this is used to get the user information from IPFS and the blockchain controller.
16) Transactions take place when there is interaction with the IPFS. All transaction address hashes are saved here.
17) Only stores the CV-ID and the IPFS hash.
18) Blockchain servers are the dedicated nodes which contain the protocol, rules and database related to the blockchain.
19) On the blockchain itself, for security all transaction data is encoded, and no CV-ID is stored here. Multiple public and private keys are used before these transaction addresses are being pushed to the blockchain node server.

    Blockchain contains the historical transaction ID (hash code).

    Please note: the same clone of the database is synced across all node servers. The CV-ID is the wallet address, so it is stored in the accounts flag of the Ethereum blockchain.

    There is no deciding parameter for miner node as per the geth node configuration but as per the server configuration the miner node should have high processing power to mine all the transactions.

    The design of the technical architecture has never envisaged that there would ever be a single node server.

IPFS (Inter Planetary File System)

20) When the user registration process is done, then this is where the information is being saved. This includes: biometrics, documentation, photos, etc.
21) All data stored on IPFS is encoded twice before being pushed to the IPFS servers.
22) The person with the root access (IPFS hash) is the only person who can access the data. The data on the IPFS is scattered in tiny pieces, and only the person with root access (IPFS hash) can reconstruct it.
23) If the IPFS hash somehow became known, then access will not be compromised as you would additionally need to know the access ID of the dedicated IPFS server, from which the data has been pushed.
24) Root access to IPFS is different for every user, so each user has his or her own root access.

25) As stated earlier all the personal data for each user is encrypted twice before being pushed to IPFS. Encryption keys are specific to the user and can only be accessed using the user CV-ID, and user data can only be accessed when the user explicitly provides access to that data by providing biometric authentication.

    Please note: two levels of encryption have been implemented. One level uses AES encryption and the other uses BLUE-FISH encryption, however before either of these all of the data gets encrypted with BASE64.

    Encryption keys are generated with the random combination of {a-z , A-Z , 1-9} with a length of 64 and 32 characters.

    Encryption keys are linked to the CV-ID in the database by an association between the server_key table and the users table. The keys are retrieved via this database relationship.

    The IPFS protocol provides for the storage of the data in tiny pieces. For more information please refer to: https://github.com/ipfs/papers/raw/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf.

    The source code for the IPFS servers is available from the IPFS Github account and under the MIT license. For more information please refer to: https://github.com/ipfs/go-ipfs/blob/master/LICENSE.

Verification Server

26) This server takes two different biometric data sets and compares them to see if they match or not.
27) The biometric data is taken from the application server. The application server gets the user's biometric data from the blockchain controller, the blockchain controller takes the biometric data from when the user first registered. The application server takes the biometric data from the blockchain controller and from the user mobile application and compares it for authenticity.

    Please note: (as noted by Terence Poon) to ensure that enrolled images comply with ISO standards or backend processing system and are of sufficient quality to perform biometric matching, we must implement the PreFace SDK and IrisCheck SDK from Aware in version 1.2.

    Update: We have the SDKs to implement and test.

Database

28) Stores the token id (verification id), time stamp, and the user-id (CV-ID) for each specific transaction.
29) It creates a generic verification id which is a token.
30) The token is only valid for 3 minutes.
31) user_id is used in database to maintain the primary key.
32) The database timestamp is the only time at which any operation was performed on database.

Mobile App Pulls

33) Application server sends over to the mobile device a token id that is only good for 3 minutes. The token (verification) id code is given to the company to see if it is a pass or fail.
34) The user then can see the verification id on the mobile app.

Company Admin

35) Company clicks on the check code button
36) The company then inserts the code that the user gave him.
37) Then that code is verified at the backend by the application server and the user is verified.

Blockchain Controller

38) Receives the client data from the company or sends the existing client data to the company.

IPFS Server

39) A distributed file system that stores all the client data. Including images, pdf's, word documents etc.

Blockchain Server

40) Keeps a record of all the transactions. A record of each time a new client is registered, and information stored in IPFS, a record of each time updates or inserts are made to the IPFS.
41) A record of each time the company requests the user's information from blockchain controller is kept in application server database.

Database

42) Basic user identification is stored here.
43) CV-ID of the user id is stored here.
44) Timestamp.
45) A record of each time the company requests the user's information from the blockchain controller is kept in application server database will be included in version 1.2.

Appendix 1: Functional roadmap

Roles [version1.2]

We need to add new roles to the solution to meet the demand for functionality as the solution is deployed.

46) Administrator role needs to be split into two:
    a. Company registrar who sets up the administrator users in a company as well as setting the minimum biometric standards that the Ecosystem / Company is prepared to accept.
    b. Normal administrators who manage the user onboarding process and day-to-day functionality.
47) Users role need to be split into two:
    a. Verified 'golden' users who have attended an authorised attestation office where they have given their full biometrics and had their legal documents certified for KYC / AML purposes.
    b. Normal users who have self-supplied their biometric data for use in non-KYC / AML environments.
48) Authenticator role. We have envisioned a new third-party proof ID with our prospects as there are instances where a third party will want to deploy personnel who can vet users via the CrossVerify application are who they claim to be. Only users with the authenticator role would have access to this functionality.

Pull biometric data [version 1.2]

49) We need to be able to pull biometric from a stationary device in a consulate or office.

Verification counter [version 1.2]

50) For every authentication we need a counter to see usage details. The counter needs to track fails as well.

UI Aesthetics and navigational improvements [version 1.2]

51) There are a number of simplifications and automations that need to be implemented.

Biometric login as an SDK [version 1.3]

52) There are many opportunities to use CrossVerify to automate application and website login.

Peer to peer [version 1.3]

53) It is important that users can identify that another person is who they say there are, either in a financial or social context. This works both in a bi-directional and one-way manner.

Appendix 2: Technical roadmap

Biometric enhancements [version 1.2]

54) Aware's Nexa|Face and Nexa|Iris solutions are in the solution today. We are implementing PreFace and IrisCheck to improve the quality of the biometric capture to the highest ISO levels.

We have implemented 1, we are starting to implement 2, and we need to evaluate 3.

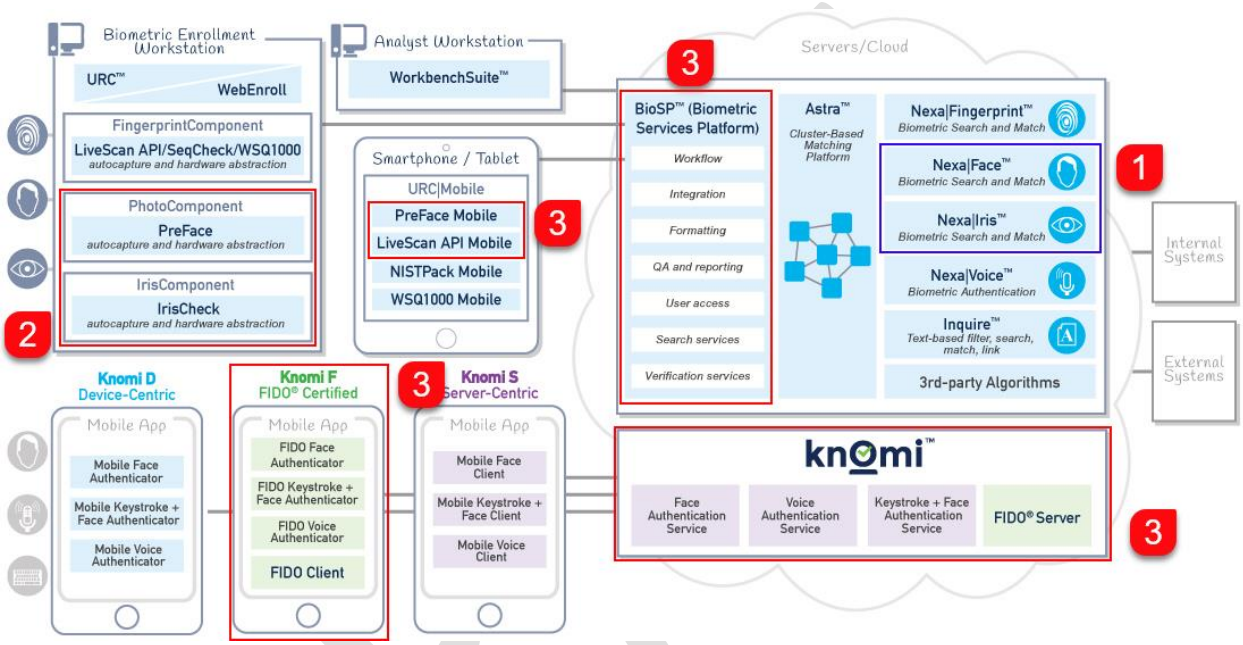


Figure 4 Aware Inc Biometrics Software Products

Blockchain enhancements [version 1.3]

55) R3's Corda has just been released and this offers greater privacy, security and scalability to CrossVerify compared to Ethereum. Given our target market a partnership with R3 would make strategic sense.

Appendix 3: Technical details

User Mobile/Cell

56) OS Version Support of Android and IOS:
    a. Android: L, M and N. Minimum SDK version: 21, target SDK version: 25.
    b. iOS: 8, 9 and 10
57) Third Party Libraries:
    a. Google Mobile Vision SDK: For face and iris recognition, both iOS and Android.
    b. Zxing QR Scanning: Third party open-source library, used for QR Scanning in android.

Application Server

58) nginx version: nginx/1.10.3 (Ubuntu)
59) Unicorn version: unicorn v5.1.0
60) Redis Server Version: 3.2.8
61) OS: Ubuntu 16.04.1 LTS
62) Processor - Intel(R) Xeon(R) CPU E5-2666 v3 @2.90GHz
63) Ruby Version: ruby 2.2.6p396
64) Rails Version: Rails 5.0.0.1

Browser Compatibility

65) Latest Versions of Internet Explorer, Google Chrome, Safari and Firefox.

Ruby on Rails Database

66) Database: PostgreSQL 9.5.4
67) Storage: 100GB
68) Database name: CrossVerifyDB
69) Port: 5432

Blockchain Database

70) Database: PostgreSQL 9.5.4
71) Storage: 100GB
72) Database name: crossverifyblockchaincontrollerdb
73) Port: 5432

Verification Server

74) OS: Windows Server 2012 R2
75) RAM: 8 GB
76) Processor: Intel(R) Xeon(R) CPU E5-2666 v3 @2.90GHz

Blockchain Controller Server

77) OS: Ubuntu 16.04.1 LTS
78) RAM: 7 GB
79) Processor: Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz

Deployed Blockchain Controller

80) 10.50.3.187
81) 10.50.2.165

IPFS Server

82) OS: Ubuntu 16.04.1 LTS
83) RAM: 3 GB
84) Processor: Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz

Deployed IPFS Nodes

85) 35.154.219.33
86) 35.154.186.151
87) 35.154.217.27
88) 35.154.247.100
89) 35.154.209.5
90) 35.154.247.222

Blockchain Node Server Details:

91) OS: Ubuntu 16.04.1 LTS
92) RAM: 7 GB
93) Processor: Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz

Deployed Private IPs

a. 10.50.2.98
b. 10.50.3.66
c. 10.50.3.147
d. 10.50.3.88
e. 10.50.2.205

Bastion Server to access the private IPs

Deployed IPv4 Public IP

94) CrossVerifyVPCStack-public1-bastion: 35.154.236.157
95) CrossVerifyVPCStack-public2-bastion: 52.66.70.211

Blockchain Explorer

96) OS: Ubuntu 16.04.2 LTS
97) RAM: 3 GB
98) Processor: Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz

Deployed BlockChain Explorer

99) CrossVerifyBlockchainExplorer: 52.66.190.234